Objective and Motivation

Our objective is to accurately classify different accents of English speakers based on short time samples. This idea is a challenge compared to language classification, which would have sounds unique to each language to ease the classification problem. If successful, this would be an important step in speech recognition -- as in, knowing the speaker’s accent would aid in recognizing the words spoken. For example, a speech recognition system could calibrate its algorithm by first using an accent classifier. Being able to understand accented speakers is critical for speech recognition systems, which are being used more and more in technology today.

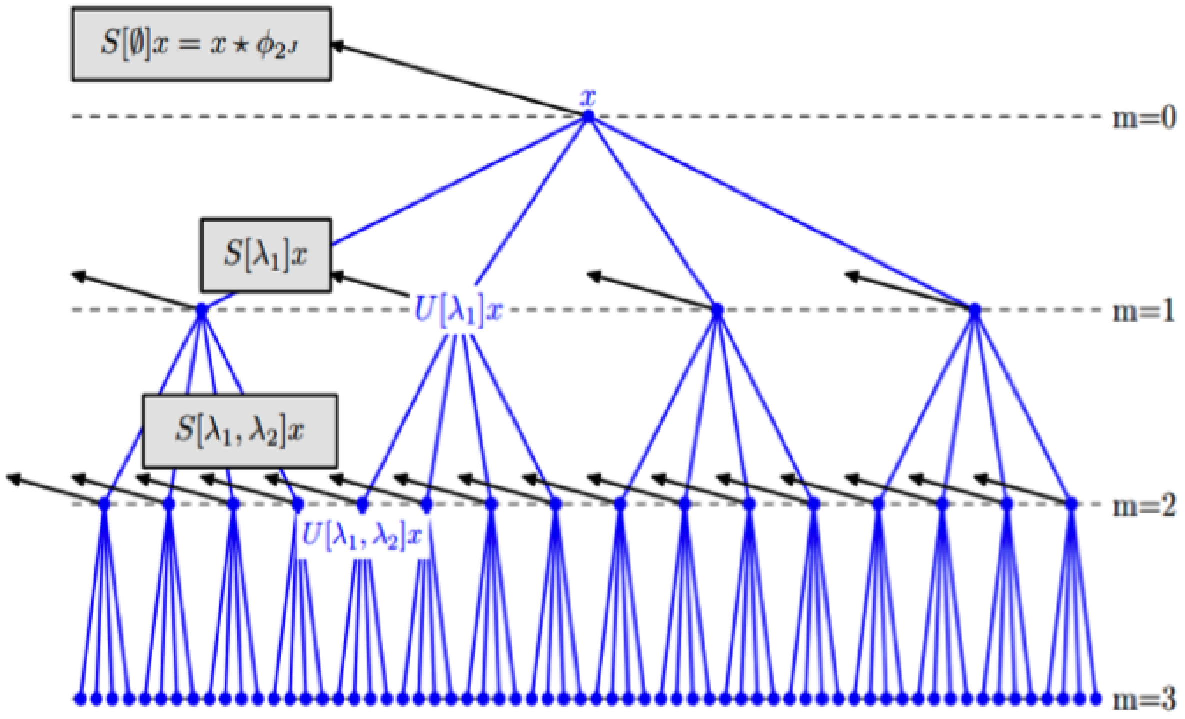
Using an online database of accents, we obtained training and testing data sets. From this database, we chose five accents to work with that had a sizable amount of available samples -- Arabic, French, Spanish, Mandarin, and English, specifically American English (4). This selection also covered many of the most common languages in the world. Armed with these audio files categorized by accent, we were able to go about coding an accent classifier.

Previous studies have typically tested one or two accents, such as only identifying Shanghai-accented Mandarin or distinguishing American English from British English (5,6). Methods often include specifically studying phoneme information, such as the study on Mandarin (5). Another study examined three different accents classified with an artificial neural network. This study had about a 70% accuracy rating, but the algorithm only classified on male voices and threw out samples that it was unsure about (7). We wanted to use a different type of classification in order to bypass studying the difference between specific phonemes in accented speakers, to not have to classify by gender, and to be able to classify between multiple accents. Therefore, we chose to use a scattering coefficient network to extract features, followed by support vector machines to classify those features. The following sections explain both of these concepts and our reasoning for using them, as well as the details of our approach.

Background Information

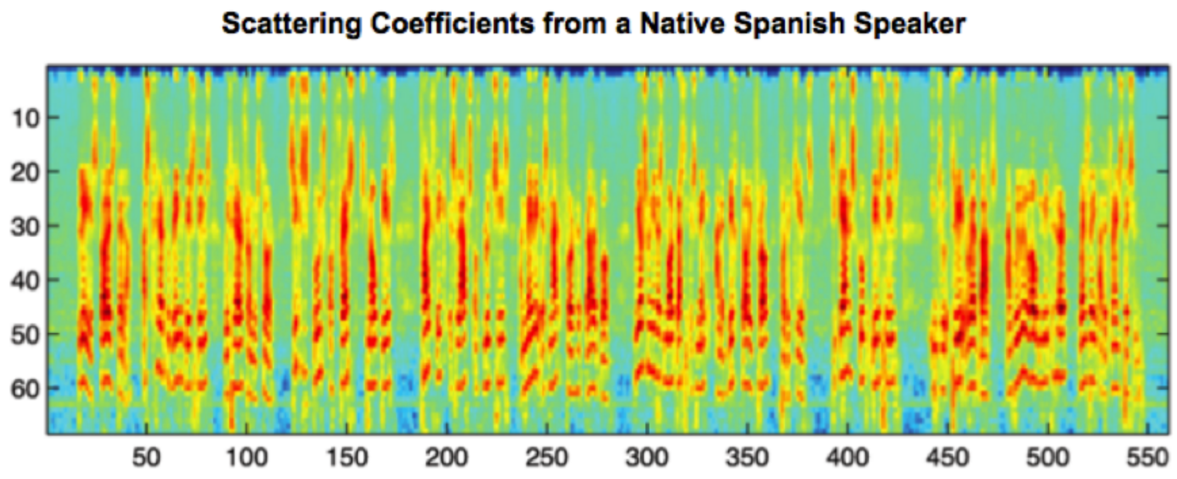
Scattering Coefficients

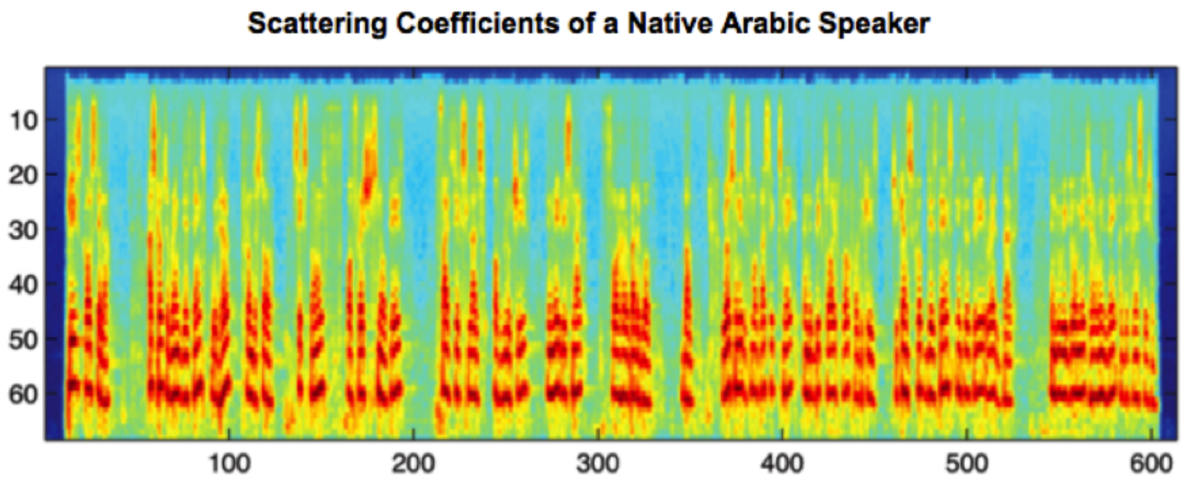
Scattering networks compute a representation of a signal that preserves high frequency information for classification. Scattering networks have been shown to be successful in classifying images in texture analysis and in handwritten digit analysis, and we believe that this type of transformation would be equally successful in audio classification. We chose to use a scattering coefficient network because a scattering transform is stable to deformations within the classes. By this we mean that even when different characteristics of the accent samples, such as gender or speed of speech, vary, the representation is not affected (2).



$S[\emptyset]x = x \star \phi_{2^J}$
x
m=0
$S[\lambda_1]x$
$U[\lambda_1]x$
m=1
$S[\lambda_1, \lambda_2]x$
m=2
$U[\lambda_1, \lambda_2]x$
m=3

The stability of scattering transforms is due to using wavelet transform coefficients and non-linear operators. The wavelet transform is a form of time-frequency representation and is similar to the Fourier transform, except that its bases are wavelets instead of sinusoids. The scattering network, as shown above, “cascades wavelet transform convolutions with non-linear modulus and averaging operators” (2). A scattering propagator applied to the signal computes the first layer signals U[λ1]x at m=1. Applying the propagator again outputs the first order scattering coefficients S[λ1]x and the propagated signal of the second layer at m=2, and so on (2).





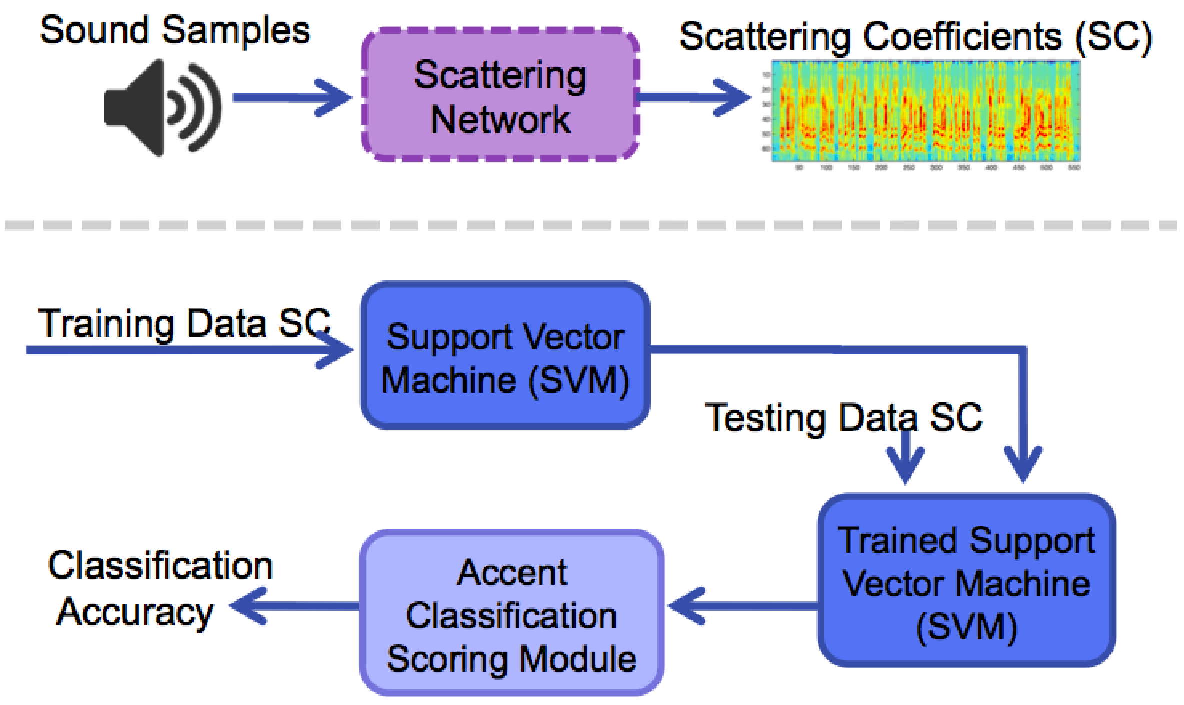
The first layer of scattering coefficients can be shown visually as a time-frequency representation called a ‘scattergram’ (1). Similar to a spectrogram, the horizontal axis is time and the vertical axis is frequency. As shown above, the scattergrams of two male voices, speaking the same English words in different accents look distinctly different. It can be seen in this example that the Arabic accent has much more high frequency content than the Spanish accent.

Support Vector Machines

Once we obtained the scattering coefficients, we vectorized the data and used Support Vector Machines (SVM) to classify them. A support vector machine is a fairly common machine learning algorithm that is used to classify an unknown vector into one of two categories. The support vector machine is first trained from data in the vectors from two separate vector sets. The algorithm finds the hyperplane that separates the two datasets with the largest margin in between the two. The support vectors are those vectors that make up the margin of each dataset -- that is, the vectors that are the closest to the hyperplane. The algorithm can then categorize data by looking at which set of support vectors the input is closer to (3).

In our application, each SVM is trained with datasets from two accents, and then the SVM classifies the testing examples in one of these two categories. About 80% (48 samples) of the sample data for each accent were used to train and 20% (12 samples) were used to test the SVMs. Because SVM works for binary classification and we have five different accent types, we devised a voting mechanism to accurately classify between the five accents. An SVM is run for every combination of two accents, for a total of 5 choose 2 = 10 SVMs. Each SVM outputs a ‘score,’ showing how likely each sample is to be one of the two accents. After all the SVMs are run, the sample is classified into the accent with the highest combined score.

Classification Dataflow


Sound Samples
Scattering Network
Scattering Coefficients (SC)
Training Data SC
Support Vector Machine (SVM)
Testing Data SC
Trained Support Vector Machine (SVM)
Accent Classification Scoring Module
Classification Accuracy

Our system is composed of two main processes: first we use a scattering network to extract the scattering coefficients, and then we train and run a support vector machine to classify the scattering coefficients. We begin with sound samples containing non native speakers reading the same paragraph aloud. A pair of two-second chunks — snippets of the sound sample — are extracted from the beginning of each sample and processed with the Matlab function ScatNet to extract a vector containing scattering coefficients.

These vectors are used to train a support vector machine Matlab function to recognize key characteristics of scattering coefficients of each accent. Scattering coefficients of testing data — obtained using the same method outlined previously — are then processed with the trained support vector machine. The support vector machine compares two accents at a time, and outputs scores between -1 and 1 of how close each sample is to one of the two accents. These scores are then processed with an accent classification scoring module to decide which accent out of all five each testing sample is closest to, and calculate the overall classification accuracy of the system.

Algorithm Code

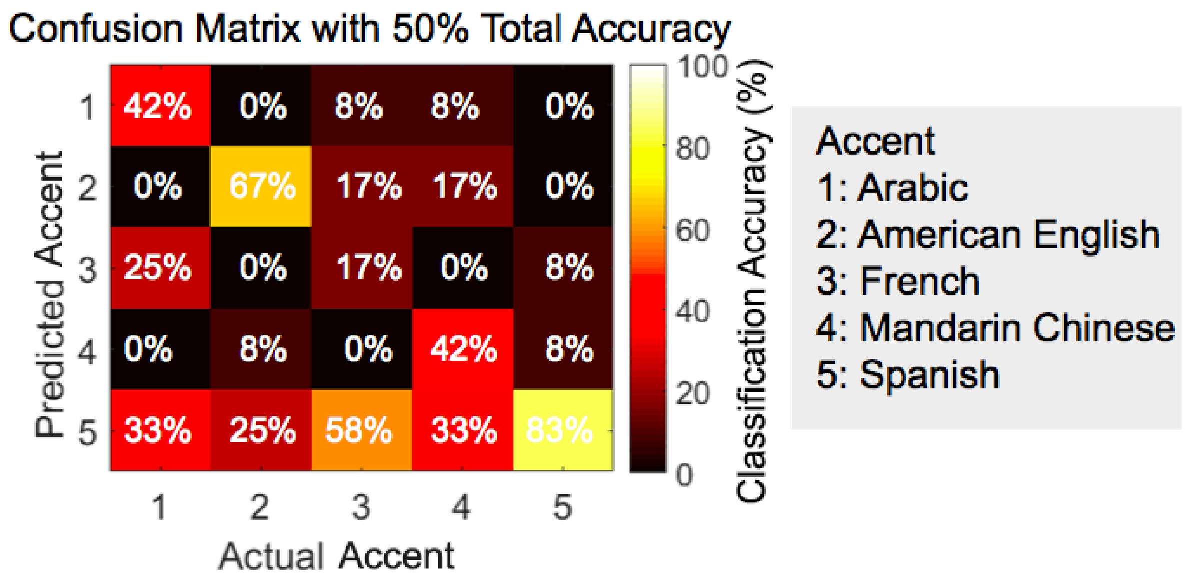
We organized our code into one main function that handles all of the SVM classifications, and a helper function that takes as input a folder of audio files and spits out the vectorized scattering coefficients for each one, stacked on top of each other in one big matrix that we can feed into the SVM. The function also takes in parameters that determine the number of chunks we split each file into and the length of each chunk, as well as some other inputs for selecting the right files from the right folder. The main function first calls this helper function on all of our accents (training and testing data) to generate all of the scattering coefficients. We then loop through all 10 pairs of accents and create 10 SVMs from their respective training matrices; the testing matrices are stacked together into one big matrix. Each SVM takes two parameters in its construction - a box constraint and a kernel scale, as well as an option for kernel function. These parameters basically tune the SVM to better categorize the specific types of vector it is given. The main function then tests these SVMs with the testing matrix, looping through each SVM and adding up the scores of the winner of each pair, then picking the accent with the highest total. Chunks from the same testing file are added to the same running total, so that one decision is made per audio file. We now have a list of guesses for each audio file, and we know what accent each file actually is. The main function organizes this data into a confusion matrix, and computes the total accuracy and the accuracy of the worst-classified accent.

Optimization
To fine tune our system, we ran large chunks of our code with varied parameters to determine what parameters make our system the best at classifying audio files. First, we tuned the input to our scattering networks by varying the number of chunks we split each of our files into and the length of each chunk. We ran our entire program with 1,2,3, and 4 chunks, and between 1 and 7 seconds per chunk (excluding values that exceed the file length). We used a gaussian kernel for our SVM with the default parameters for this step. Our results showed high accuracies when we broke our signal into two chunks, and also when the total length equaled four seconds. Based off of this, we picked two chunks of two seconds each to compute our optimized scattering coefficients. We then ran these optimized

coefficients through the rest of our system, this time varying the two parameters for our SVM - kernel scale and box constraint, as well as trying gaussian, polynomial, and linear kernels. We chose a gaussian kernel with parameters that yielded a high total accuracy and a high accuracy of the worst classified accent.

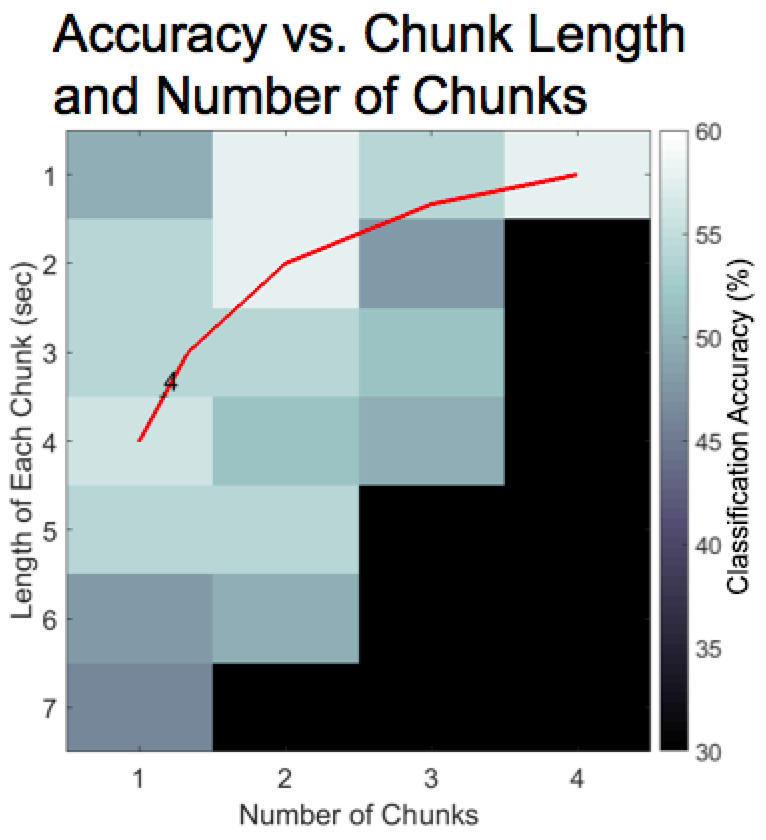
The code used for optimization be viewed here.

Accuracy of Results



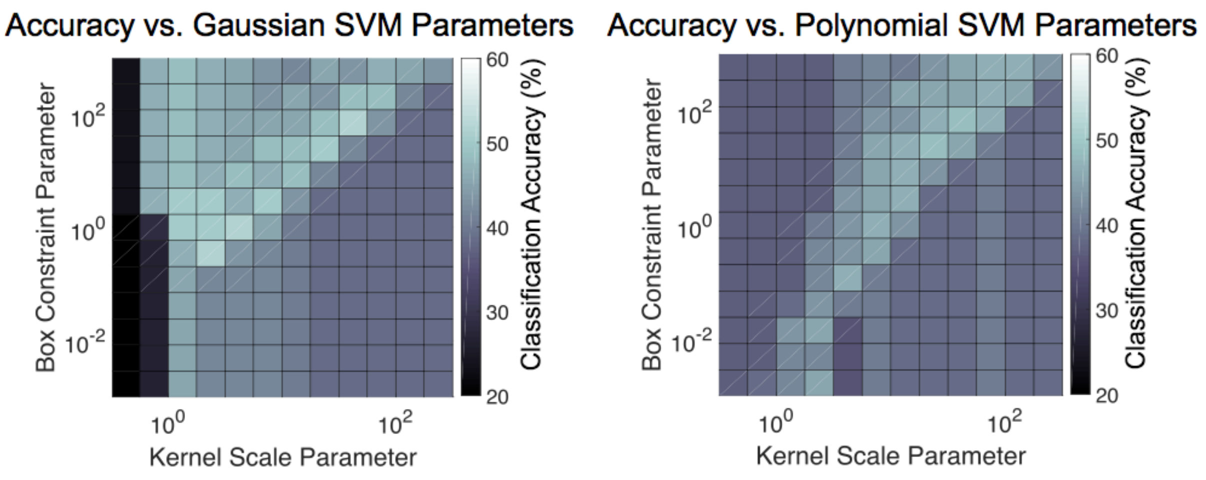
A confusion matrix detailing the results of a testing run.

Confusion matrices are used to analyze the results of the full testing runs. A confusion matrix, also known as an error matrix, gives a visual representation of the performance of a learning algorithm--in our case, using scattering coefficients to train an SVM. Each entry in the matrix corresponds to the predicted accent versus the actual accent, with entries along the diagonal representing correctly predicted accents. While no column contained perfect classification accuracy, we were still able to correctly classify four out of five of the accents. We found that the overall accuracy of the confusion matrix was 50%, with a high misclassification of French as Spanish. We found an overall tendency for languages to be classified as Spanish, which we believe is due to the diversity of the Spanish accent sound samples. The accent database had fewer Spanish accent samples than the other four languages so the samples included speakers from many Spanish speaking countries. The diversity in the Spanish accent sound samples might have led to the high rates of misclassification as Spanish observed.



Results of tuning the parameters of the input data.

When tuning the input data parameters — the number of chunks and the length of each chunk — we found that, in general, having two total chunks resulted in the best performance. Furthermore, the system performed well when the total time of the sound analyzed — the length of each chunk times the number of chunks — was four seconds, a trend which is highlighted in red on the graph. We hypothesize that this occurs because the sound clips are relatively synchronized up until four seconds, with the pace of the speaker being increasingly significant afterwards. We applied these results to later tests by using a pair of two second chunks as the input data parameters.



Results of tuning the parameters of the SVM.

Through testing combinations of box constraint and kernel scale parameters for linear, gaussian, and polynomial support vector machine models, we were able to determine the optimal parameters for classification accuracy. We found that the linear model produced far worse results than the gaussian and polynomial models and eliminated it from further analysis. The gaussian model had better overall accuracy compared to the polynomial model, so we chose our ideal testing parameters to be in the range of its lightest squares — corresponding to about 50% accuracy. These parameters were used to produce the confusion matrix seen earlier.

Conclusion and Future Applications

Conclusion

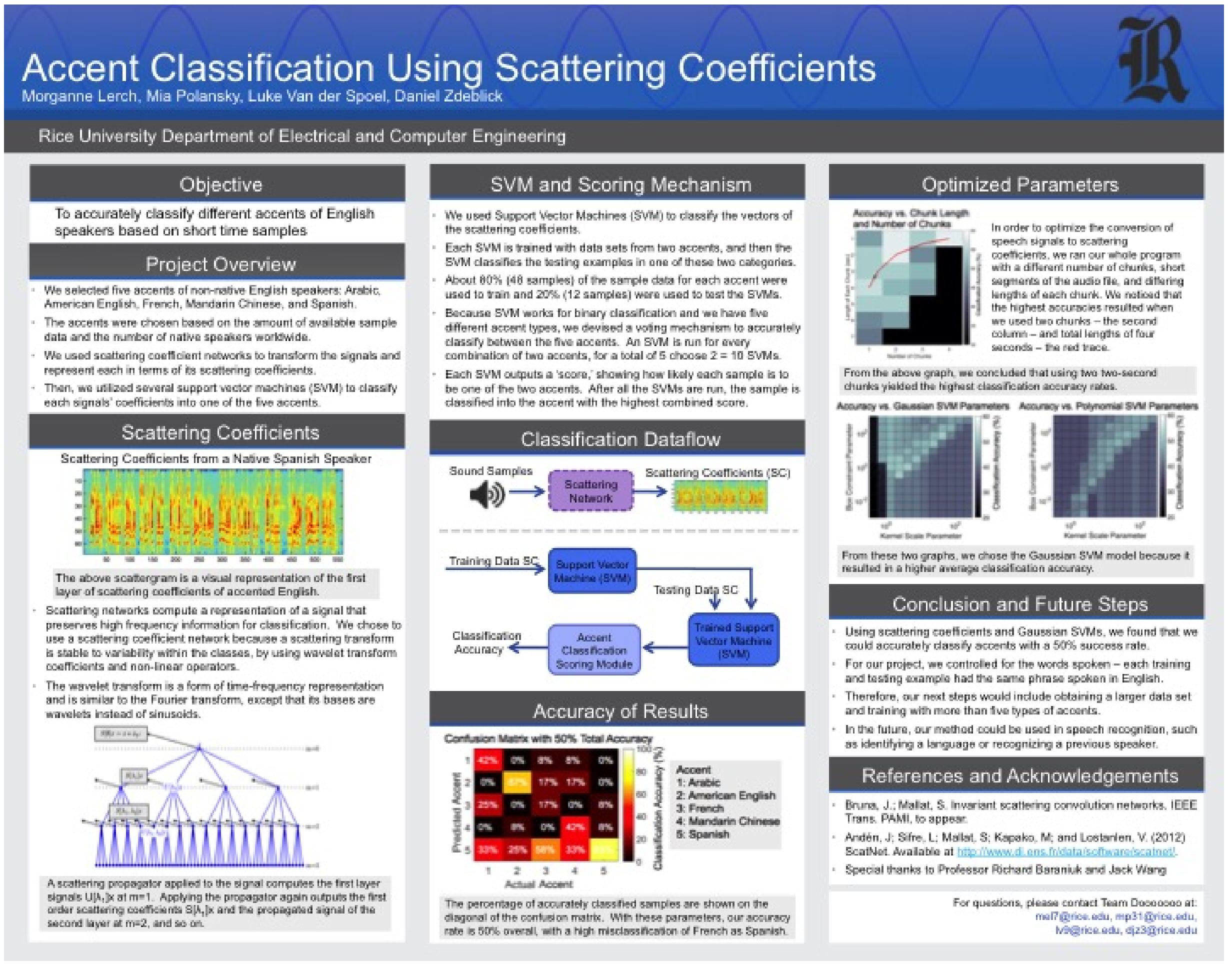
By optimizing our parameters, we could categorize two two-second speech samples into five accents with an accuracy of 50% or greater, compared to an accuracy of 20% for randomly guessing. This is quite an impressive feat considering the variability in people's voices and that humans themselves have trouble recognizing accents with only a two-second sample. Our algorithm tended to misclassify accents as Spanish -- we hypothesized that this is an artifact of the wide range of origin of the Spanish speakers in our data set. We hope to fix this by using a more complete data set in which we have more training samples for different regional accents.

Potential Applications

Categorizing accents can be used to improve speech recognition. For example, if the program realizes that it is listening to a certain accent, it could adjust its algorithm to better register the words spoken. Improving the accuracy of speech recognition systems is very important these days, considering their prevalence. Some further applications of using scattering coefficients to categorize speech samples might be recognizing a previous speaker in a recording or classifying the language that someone is speaking.

Future Work

In the future we can work on improving our algorithm by optimizing the parameters for the scattering coefficients, which were pre-built into the open-source ScatNet code. In addition, training with larger and more varied data sets should lead to an increase in accuracy and an increase in the amount of accents that can be identified.



Accent Classification Using Scattering Coefficients

Morganne Lerch, Mia Polansky, Luke Van der Spoel, Daniel Zdeblick

Rice University Department of Electrical and Computer Engineering

Objective

To accurately classify different accents of English speakers based on short time samples

Project Overview

- We selected five accents of non-native English speakers: Arabic, American English, French, Mandarin Chinese, and Spanish.
- The accents were chosen based on the amount of available sample data and the number of native speakers worldwide.
- We used scattering coefficient networks to transform the signals and represent each in terms of its scattering coefficients.
- Then, we utilized several support vector machines (SVM) to classify each signals' coefficients into one of the five accents.

Scattering Coefficients

The above scattergram is a visual representation of the first layer of scattering coefficients of accented English.

- Scattering networks compute a representation of a signal that preserves high frequency information for classification. We chose to use a scattering coefficient network because a scattering transform is stable to variability within the classes, by using wavelet transform coefficients and non-linear operators.
- The wavelet transform is a form of time-frequency representation and is similar to the Fourier transform, except that its bases are wavelets instead of sinusoids.

A scattering propagator applied to the signal computes the first layer signals $U[\lambda_1]x$ at m=1. Applying the propagator again outputs the first order scattering coefficients $S[\lambda_1]x$ and the propagated signal of the second layer at m=2, and so on.

SVM and Scoring Mechanism

- We used Support Vector Machines (SVM) to classify the vectors of the scattering coefficients.
- Each SVM is trained with data sets from two accents, and then the SVM classifies the testing examples in one of these two categories.
- About 80% (48 samples) of the sample data for each accent were used to train and 20% (12 samples) were used to test the SVMs.
- Because SVM works for binary classification and we have five different accent types, we devised a voting mechanism to accurately classify between the five accents. An SVM is run for every combination of two accents, for a total of 5 choose 2 = 10 SVMs.
- Each SVM outputs a 'score,' showing how likely each sample is to be one of the two accents. After all the SVMs are run, the sample is classified into the accent with the highest combined score.

Classification Dataflow

Sound Samples
Scattering Network
Scattering Coefficients (SC)
Training Data SC
Support Vector Machine (SVM)
Testing Data SC
Trained Support Vector Machine (SVM)
Accent Classification Scoring Module
Classification Accuracy

Accuracy of Results

Confusion Matrix with 50% Total Accuracy

Predicted Accent \ Actual Accent	1	2	3	4	5
1	42%	0%	8%	8%	0%
2	0%	67%	17%	17%	0%
3	25%	0%	17%	0%	8%
4	0%	8%	0%	42%	8%
5	33%	25%	58%	33%	83%

Accent
1: Arabic
2: American English
3: French
4: Mandarin Chinese
5: Spanish

The percentage of accurately classified samples are shown on the diagonal of the confusion matrix. With these parameters, our accuracy rate is 50% overall, with a high misclassification of French as Spanish.

Optimized Parameters

In order to optimize the conversion of speech signals to scattering coefficients, we ran our whole program with a different number of chunks, short segments of the audio file, and differing lengths of each chunk. We noticed that the highest accuracies resulted when we used two chunks – the second column – and total lengths of four seconds – the red trace.

From the above graph, we concluded that using two two-second chunks yielded the highest classification accuracy rates.

Accuracy vs. Gaussian SVM Parameters
Accuracy vs. Polynomial SVM Parameters
Box Constraint Parameter
Kernel Scale Parameter
Classification Accuracy (%)

From these two graphs, we chose the Gaussian SVM model because it resulted in a higher average classification accuracy.

Conclusion and Future Steps

- Using scattering coefficients and Gaussian SVMs, we found that we could accurately classify accents with a 50% success rate.
- For our project, we controlled for the words spoken – each training and testing example had the same phrase spoken in English.
- Therefore, our next steps would include obtaining a larger data set and training with more than five types of accents.
- In the future, our method could be used in speech recognition, such as identifying a language or recognizing a previous speaker.

References and Acknowledgements

- Bruna, J.; Mallat, S. Invariant scattering convolution networks. IEEE Trans. PAMI, to appear.
- Andén, J; Sifre, L; Mallat, S; Kapako, M; and Lostanlen, V. (2012) ScatNet. Available at http://www.di.ens.fr/data/software/scatnet/.
- Special thanks to Professor Richard Baraniuk and Jack Wang

For questions, please contact Team Dooooooo at:
mel7@rice.edu, mp31@rice.edu,
lv9@rice.edu, djz3@rice.edu

References

(1) Andén, J; Sifre, L; Mallat, S; Kapako, M; and Lostanlen, V. (2012) ScatNet. (2) Bruna, J.; Mallat, S. Invariant scattering convolution networks. IEEE Trans. PAMI, to appear. (3) "Support Vector Machines (SVM)." MATLAB Documentation. MathWorks, n.d. Web. 16 Dec. 2015. (4) Weinberger, Steven. The Speech Accent Archive. George Mason University, n.d. Web. 16 Dec. 2015. (5) Zheng, Yanli, et al. "Accent detection and speech recognition for Shanghai-accented Mandarin." Interspeech. 2005. (6) Lin, Xiaofan, and Steven Simske. "Phoneme-less hierarchical accent classification." Signals, Systems and Computers, 2004. Conference Record of the Thirty-Eighth Asilomar Conference on. Vol. 2. IEEE, 2004. (7) Blackburn, C. S., Julie Vonwiller, and R. W. King. "Automatic accent classification using artificial neural networks." Eurospeech. 1993.